Partial Logic Synthesis

Synthesis of ECO & Rectification Functions

Priyank Kalla

Professor
Electrical and Computer Engineering, University of Utah
kalla@ece.utah.edu
`http://www.ece.utah.edu/~kalla`

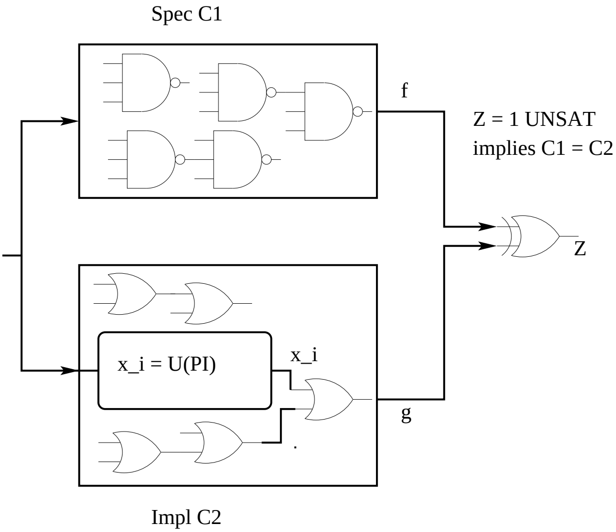
Problem Description

- Given: *Spec* function f or a gate-level circuit C_1
- Given: Implementation *Impl* function g or a gate-level circuit C_2
- Suppose *Spec* $\neq$ *Impl*, i.e. Equivalence checking between C_1, C_2 generates a counter-example
- We have to rectify C_2 to match the *Spec*, but do not resynthesize the whole circuit C_2
  - Perform *single-fix* or *multi-fix* rectification
  - Identify a set of nets $\{x_i\}$ in C_2 where rectification can be applied
  - Compute a rectification function U, replace $x_i = U$, such that $C_1 \equiv C_2$

The Scenario

- Compute rectification patches for buggy implementations C_2
- Compute rectification patches for Engineering Change Orders (ECO)
  - Suppose, given C_1 *Spec* and correct *Impl* C_2
  - Make slight modifications on C_1: small change in functionality
  - New $C_1 \neq C_2$ (bug!)
  - Minimally modify C_2 to match the *Spec*
- Topologically constrained logic synthesis, synthesize subcircuits partially

The Scenario: Miter Model



Single-Fix Rectification

- Irrespective of the type of bugs, or the number of error inputs (counter-examples) ...
- Does there exist a net x_i and a Boolean function $U(X_{PI})$, s.t. $x_i = U(X_{PI})$ makes $C_1 \equiv C_2$?
- Mathematically, a Quantified Boolean Formula:
  - $\exists U, \forall X_{PI} \quad C_1 \equiv C_2$ is SAT
- Single-fix rectification may or may not exist
- Multi-fix: Find $x_1 = U_1, x_2 = U_2, \ldots, x_t = U_t$, s.t. $C_1 \equiv C_2$

Single-Fix Rectification

The Problem is three-fold:

- Once C_1 does not match C_2
- Identify a net x_i where single-fix rectification is admissible. So how to pick the net x_i? (Discuss later...)
- How to ascertain that there exists a single-fix rectification function $x_i = U(X_{PI})$? (Decision procedure)
- If single-fix exists at x_i, find the Boolean function U. (Quantification procedure).

Existence of Single-fix at net x_i

Theorem (Existence of Single-fix rectification at net x_i (Thm. 1))

Given the Spec circuit C_1 with function f and Impl circuit C_2 with function g, where X is the set of primary inputs, $f \neq g$, and a net $x_i \in C_2$, the circuit C_2 can be single-fix rectified at x_i with function $x_i = U(X_{PI})$, **if and only if**

$$[f(X) \oplus g(X, x_i = 0)] \ \wedge \ [f(X) \oplus g(X, x_i = 1)] = \bot \ (UNSAT)$$
$$M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) = \bot$$

Existence of Single-fix at net x_i

Theorem (Existence of Single-fix rectification at net x_i (Thm. 1))

Given the Spec circuit C_1 with function f and Impl circuit C_2 with function g, where X is the set of primary inputs, $f \neq g$, and a net $x_i \in C_2$, the circuit C_2 can be single-fix rectified at x_i with function $x_i = U(X_{PI})$, **if and only if**

$$[f(X) \oplus g(X, x_i = 0)] \ \wedge \ [f(X) \oplus g(X, x_i = 1)] = \bot \ (UNSAT)$$
$$M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) = \bot$$

What are $[f(X) \oplus g(X, x_i = 0)]$ and $[f(X) \oplus g(X, x_i = 1)]$?

Existence of Single-fix at net x_i

Theorem (Existence of Single-fix rectification at net x_i (Thm. 1))

Given the Spec circuit C_1 with function f and Impl circuit C_2 with function g, where X is the set of primary inputs, $f \neq g$, and a net $x_i \in C_2$, the circuit C_2 can be single-fix rectified at x_i with function $x_i = U(X_{PI})$, **if and only if**

$$[f(X) \oplus g(X, x_i = 0)] \ \wedge \ [f(X) \oplus g(X, x_i = 1)] = \perp \ (UNSAT)$$
$$M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) = \perp$$

What are $[f(X) \oplus g(X, x_i = 0)]$ and $[f(X) \oplus g(X, x_i = 1)]$?

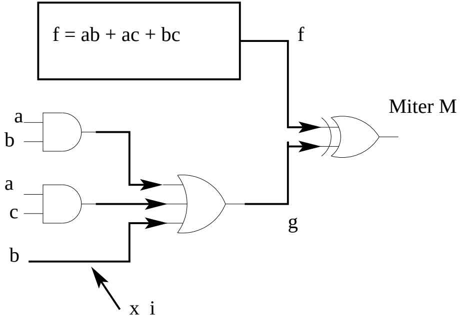
The set of all test vectors for x_i stuck-at-0, and x_i stuck-at-1, resp.
In other words, $[f(X) \oplus g(X, x_i = 0)]$ = primary input assignments to X (minterms) that differentiate $f(X)$ from $g(X, x_i = 0)$.

Single-Fix Rectification Test

- $M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) = \perp$ means that **product of positive and negative co-factors** of the miter w.r.t. x_i is empty.
- If $M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) \neq 0$, then: there exists a minterm X s.t. the difference between *Spec* and *Impl* is observed for both values of target net x_i
- Equivalently, if $M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) \neq 0$, no matter what value x_i takes for this minterm X, functional difference between *Spec* and *Impl* remains.
- In other words, rectification is only feasible when $M(X, x_i = 0) \ \wedge \ M(X, x_i = 1) = 0$.

Example

- Let $f = ab + ac + bc$ and $g = ab + ac + b$
- Clearly, $f \neq g$, can g be rectified at net x_i shown below?



Rectification Check Passes: Example

In the example on the previous slide:

- Let $f = ab + ac + bc$ and $g = ab + ac + b$
- $M(X, x_i = 0) = (ab + ac + bc) \oplus (ab + ac + 0) = a'bc$
- $M(X, x_i = 1) = (ab + ac + bc) \oplus (ab + ac + 1) = a'b' + a'c' + b'c'$
- $M(X, x_i = 0) \wedge M(X, x_i = 1) = 0$, so rectification is feasible

Rectification Check Fails: Example

- Let $f = ab + ac + bc$ (spec) and $g = a + b$ (buggy implementation)
- Check for single-fix rectifiability at $x_i = b$ (input of OR gate) in g
- $M(X, x_i = 0) = (ab + ac + bc) \oplus (a + 0) = a'bc + ab'c'$
- $M(X, x_i = 1) = (ab + ac + bc) \oplus (a + 1) = a'b' + a'c' + b'c'$
- $M(X, x_i = 0) \wedge M(X, x_i = 1) = ab'c' \neq 0$, so rectification at the b input of the OR gate in g is not possible.

Compute Single-Fix Rectification Function $U(X_{PI})$

Theorem (Compute Rectification function $x_i = U(X_{PI})$)

When the above condition (Theorem 1) is satisfied, then the single-fix rectification function can be computed as:

$$[f(X) \oplus g(X, x_i = 0)] \subseteq U(X) \subseteq \overline{[f(X) \oplus g(X, x_i = 1)]}$$

$$M(X, x_i = 0) \subseteq U(X) \subseteq \overline{M(X, x_i = 1)}$$

Compute Single-Fix Rectification Function $U(X_{PI})$

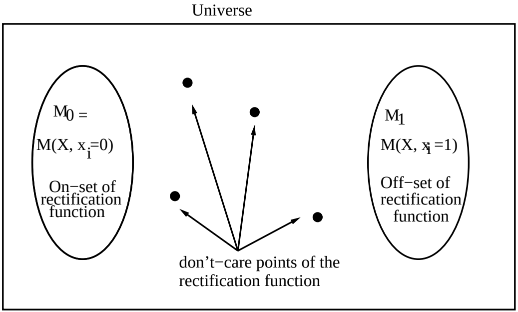
Theorem (Compute Rectification function $x_i = U(X_{PI})$)

When the above condition (Theorem 1) is satisfied, then the single-fix rectification function can be computed as:

$$[f(X) \oplus g(X, x_i = 0)] \subseteq U(X) \subseteq \overline{[f(X) \oplus g(X, x_i = 1)]}$$
$$M(X, x_i = 0) \subseteq U(X) \subseteq \overline{M(X, x_i = 1)}$$

- $[f(X) \oplus g(X, x_i = 0)] = M(X, x_i = 0) =$ ON-set of the rectification function. Call this M_0.
- $[f(X) \oplus g(X, x_i = 1)] = M(X, x_i = 1) =$ OFF-set of the rectification function. Call this M_1.
- $\overline{[f(X) \oplus g(X, x_i = 1)]} =$ ON-set $\cup$ DC-set.
- DC-set $= \overline{M_0 + M_1}$.

Characterizing the Rectification Function $U(X_{PI})$



- $[f(X) \oplus g(X, x_i = 0)] = M(X, x_i = 0) =$ ON-set of the rectification function. Call this M_0.
- $[f(X) \oplus g(X, x_i = 1)] = M(X, x_i = 1) =$ OFF-set of the rectification function. Call this M_1.
- DC-set $= \overline{M_0 + M_1}$.

Compute Single-Fix Rectification Function $U(X_{PI})$

Theorem (Compute Rectification function $x_i = U(X_{PI})$)

When the above condition (Theorem 1) is satisfied, then the single-fix rectification function can be computed as:

$$[f(X) \oplus g(X, x_i = 0)] \subseteq U(X) \subseteq \overline{[f(X) \oplus g(X, x_i = 1)]}$$

Compute Single-Fix Rectification Function $U(X_{PI})$

Theorem (Compute Rectification function $x_i = U(X_{PI})$)

When the above condition (Theorem 1) is satisfied, then the single-fix rectification function can be computed as:

$$[f(X) \oplus g(X, x_i = 0)] \subseteq U(X) \subseteq \overline{[f(X) \oplus g(X, x_i = 1)]}$$

Does this remind you of Craig Interpolants?

Compute Single-Fix Rectification Function $U(X_{PI})$

Theorem (Compute Rectification function $x_i = U(X_{PI})$)

When the above condition (Theorem 1) is satisfied, then the single-fix rectification function can be computed as:

$$[f(X) \oplus g(X, x_i = 0)] \subseteq U(X) \subseteq \overline{[f(X) \oplus g(X, x_i = 1)]}$$

Does this remind you of Craig Interpolants?

$$M(X, x_i = 0) \wedge M(X, x_i = 1) = \bot$$

There exists C.I. $I(X)$ s.t.:

- $M(X, x_i = 0) \implies I(X)$;
- $I(X) \wedge M(X, x_i = 1) = \bot$;
- $I(X)$, where X = common variables of $M(X, x_i = 0)$ and $M(X, x_i = 1)$.

Rectification function $U(X) = I(X)$!!

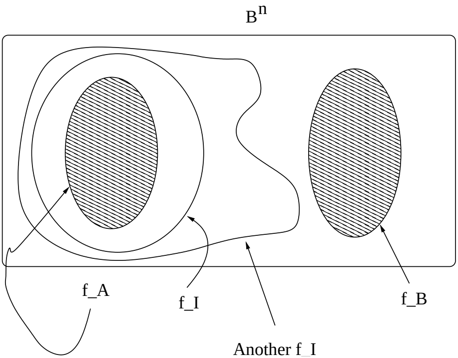
Craig Interlants: A concept of “abstraction”, for UNSAT problems

Definition

Let $f(X_A, X_B, X_C)$ be a Boolean function in variables $X = \{x_1, \dots, x_n\}$ such that X is partitioned into disjoint subsets X_A, X_B, X_C. Let $f = f_A(X_A, X_C) \wedge f_B(X_B, X_C) = \emptyset$. Then there exists another Boolean function f_I such that:

- $f_A \implies f_I$; or $f_A \subseteq f_I$
- $f_I \wedge f_B = \emptyset$
- $f_I(X_C)$ only contains X_C variables, i.e. the common variables of f_A, f_B: $Vars(f_I) \subseteq Vars(f_A) \cap Vars(f_B)$

Craig Interpolants



- The ABC tool with MiniSAT solver can return an f_I, provided f_A, f_B, X_A, X_B, X_C is given.
- Interpolant computed through a resolution proof

Craig Interpolants: Examples

There may be more than one interpolants:

- $f = f_A \cdot f_B$
- $f_A = (a_1 + a_2')(a_1' + a_3')(a_2) = a_1 a_2 a_3'$
- $f_B = (a_2' + a_3)(a_2 + a4)(a_4') = a_2 a_3 a_4'$
- $X_A = \{a_1\}, X_B = \{a_4\}, X_C = \{a_2, a_3\}$
- One interpolant $f_{I_1} = a_3' a_2$
- Another interpolant $f_{I_2} = a_3'$
- The set of all interpolants forms a **lattice**, the smallest interpolant at the bottom, and the largest at the top
- Smallest interpolant: $f_I^{smallest} = \exists_{X_A} f_A(X_A, X_C)$
- Largest interpolant: $f_I^{largest} = \overline{\exists_{X_B} f_B(X_B, X_C)}$

To compute Interpolants

Use Boolean logic manipulation, or use a SAT solver

- Using Boolean manipulation, we can compute two interpolants
- Smallest interpolant: $f_I^{smallest} = \exists_{X_A} f_A(X_A, X_C)$, where $\exists_x f(x, y, z, \dots)$ is a Boolean function defined as
  - $\exists_x f(x, y, z, \dots) = f_x + f_{\overline{x}}$, $f_x = f(x = 1)$ and $f_{\overline{x}} = f(x = 0)$.
  - f_x = the positive cofactor of f w.r.t. x, $f_{\overline{x}}$ is negative cofactor.
  - $\exists_x f(x, y, z, \dots)$ is called existential abstraction of f w.r.t. x, also called smoothing.
  - $\exists_x f(x, y, z, \dots)$ is the smallest function larger than f, i.e. it contains f and does not have x in its support.
- Largest interpolant $= \overline{\exists_{X_B} f_B(X_B, X_C)}$

Craig Interpolant: Examples

From the previous slides:

- $f_A = a_1 a_2 a_3'$ and $f_B = a_2 a_3 a_4'$
- $X_A = \{a_1\}, X_B = \{a_4\}, X_C = \{a_2, a_3\}$
- Smallest interpolant: $f_I^{smallest} = \exists_{X_A} f_A(X_A, X_C) = a_2 a_3'$
- Largest interpolant: $f_I^{largest} = \overline{\exists_{X_B} f_B(X_B, X_C)}$
  - Largest interpolant $= \overline{\exists_{X_B} a_2 a_3 a_4'} = \overline{a_2 a_3} = a_2' + a_3'$
- Let f_I be any interpolant, then $f_I^{smallest} \subseteq f_I \subseteq f_I^{largest}$

To compute Rectification Function

$$M(X, x_i = 0) \wedge M(X, x_i = 1) = \bot$$

There exists C.I. $I(X)$ s.t.:

- $M(X, x_i = 0) \implies I(X)$;
- $I(X) \wedge M(X, x_i = 1) = \bot$;
- $I(X)$, where $X =$ common variables of $M(X, x_i = 0)$ and $M(X, x_i = 1)$.

- Rectification function $x_i = U(X) = I(X)$!
- $X =$ primary inputs of the circuit, as all PIs common variables of the miter.
- $I(X) = M(X, x_i = 0)$ is the smallest interpolant, and serves as the rectification function.

Example: Compute $U(X_{PI})$

- Let $f = ab + ac + bc$ and $g = ab + ac + b$
- $U(X) = I(X) = M(X, x_i = 0) = a'bc$, the smallest interpolant works as a rectification function $g = ab + ac + a'bc$
- Largest interpolant: $\overline{M(X, x_i = 1)} = ab + ac + bc$ also works: $g = ab + ac + ab + ac + bc$
- Any interpolant works as a rectification function: $M(X, x_i = 0) \subseteq I(X) \subseteq \overline{M(X, x_i = 1)}$
  - $I(X) = bc$ also rectifies g

For Logic Synthesis of $U(X_{PI})$, synthesize: M_0 as the care-set of U, M_1 as the offset of U, and $\overline{M_1} - M_0$ as the don't care set of U.

Why $U(X_{PI}) = I(X)$ rectifies the circuit?

- $M_0 = M(X, x_i = 0)$ gives the error cubes for the implementation when $x_i = 0$
- Since $I(X) \supseteq M_0$, and $I(X)$ evaluates to 1 for all minterms of M_0, it fixes all the mismatches that occured when $x_i = 0$
- $M_1 = M(X, x_i = 1)$ gives the error cubes for the implementation when $x_i = 1$
- $I(X) \wedge M_1 = 0$, i.e. $I(X)$ evaluates to 0 for all minterms of M_1, so it fixes all the mismatches that occured when $x_i = 1$
- Thus $I(X)$ computes the rectification patch

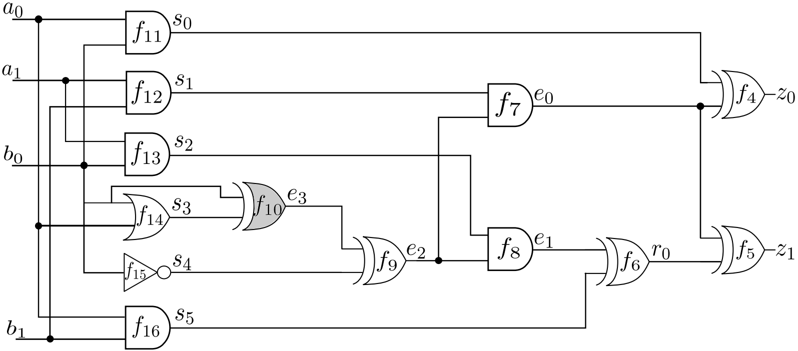
Important References [1] [2] [3] [4]

Identify Single-Fix Rectification Target Net x_i

- Perform Combinational Equivalence Checking (CEC) between C_1, C_2
  - Use the following command in ABC: `ABC> cec C1.blif C2.blif`
- Since $C_1 \neq C_2$, ABC identifies the outputs of the circuits which are affected by the bug
- Say, there exists an input (counter-example to CEC) X_1 s.t. $f_i(X_1) \neq f_j(X_1)$
- Identify the transitive fanin-cones of f_i, f_j
- Identify the set of nets $\mathcal{N}$ (gate-outputs) that lie in the intersection of fanin-cones of f_i, f_j
- Then the nets $x_i \in \mathcal{N}$ are candidates for single-fix rectification

Identify Single-Fix Target Net x_i

Example Circuit:



The circuit is buggy, bug = gate change at net e_3, which should have been an AND gate. Both outputs z_0, z_1 affected by the bug.

Identify Single-Fix Target Net x_i

- In the previous figure, e_3 should have been an AND gate in the correct circuit C_1
- Bug introduced, $e_3 = XOR$ gate
- The bug affects both outputs z_0, z_1
- Fanin cone $z_0 = \{s_0, e_0, s_1, e_2, e_3, s_3, s_4, X_{PI}\}$
- Fanin cone $z_1 = \{r_0, s_5, e_1, s_2, e_2, e_3, s_3, s_4, e_0, s_1, X_{PI}\}$
- Intersection of fanin cones: $\mathcal{N} = \{s_3, e_3, e_2, s_4, e_0, s_1, s_2, X_{PI}\}$ are targets for rectification x_i
- Select a net $x_i \in \mathcal{N}$ to see if Theorem 1 ascertains rectifiability at x_i

- C. C. Lin, K. C. Chen, S. C. Chang, and M. Marek-Sadowska, "Logic Synthesis for Engineering Change," in *Proc. Design Automation Conf. (DAC)*, 1995, pp. 647–652.
- K.-F. Tang, C.-A. Wu, P.-K. Huang, and C.-Y. R. Huang, "Interpolation based incremental ECO Synthesis for Multi-Error Logic Rectification," in *Proc. Design Automation Conf.*, 2011, pp. 146–151.
- A. Dao, N.-Z. Lee, L.-C. Chen, M. Lin, J.-H. Jiang, A. Mishchenko, and R. Brayton, "Efficient Computations of ECO Patch Functions," in *Proc. Design Auto. Conf.*, 2018.
- M. Fujita, "Toward Unification of Synthesis and Verification in Topologically Constrained Logic Design," *Proceedings of the IEEE*, 2015.